5043 Advanced Machine Learning - HW 3

Author: Enzo Durel

March 11, 2025

GALLOGLY COLLEGE OF ENGINEERING
The UNIVERSITY of OKLAHOMA

Contents

List of Figures

1 Figures

1.1 Figure 1

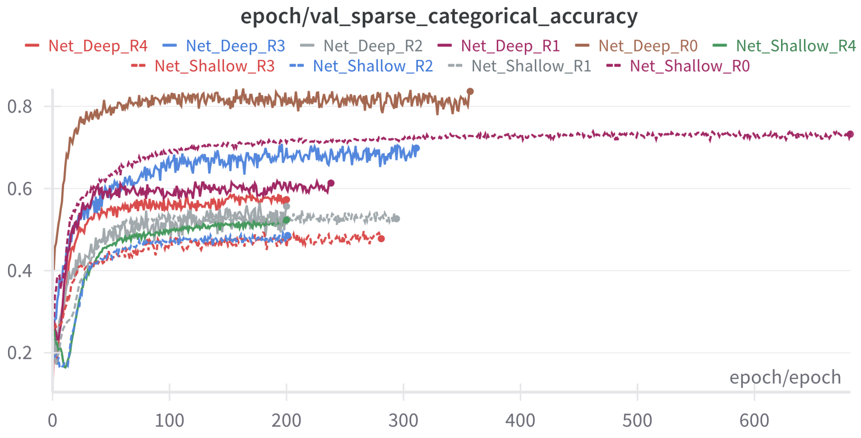

Fig. 1: Validation Accuracy as a function of epoch for the Shallow and Deep models

1.2 Figure 2

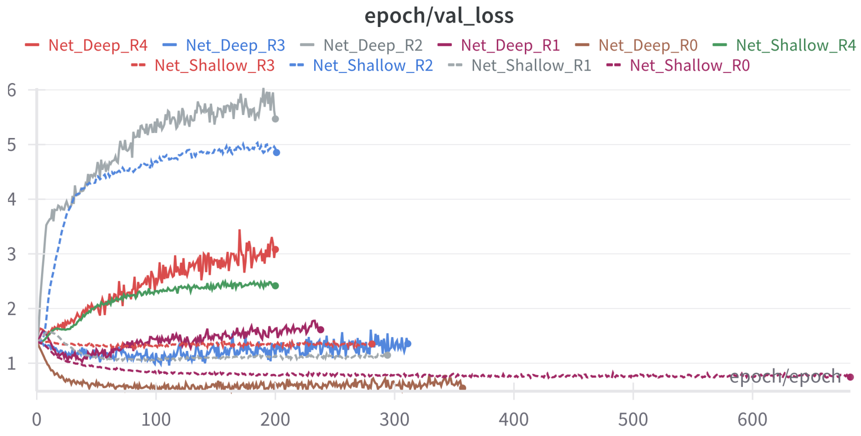

Fig. 2: Validation Loss as a function of epoch for the Shallow and Deep models

1.3 Figure 3

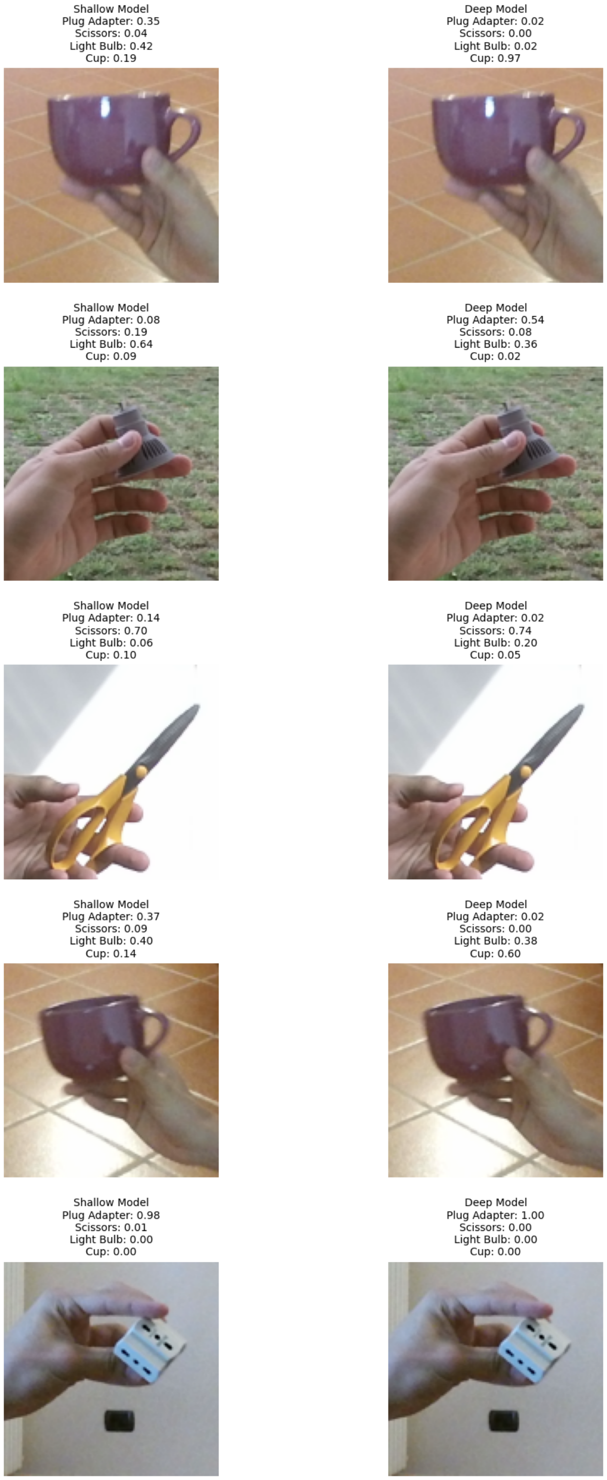

Fig. 3: Testing Sample Data Probability Distribution for the Shallow and Deep models

1.4 Figure 4a

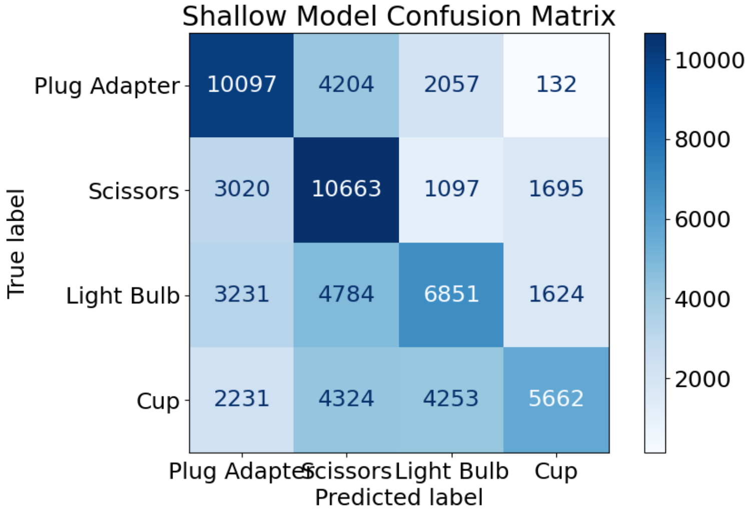

Fig. 4: Confusion Matrix of the test set data across all rotations for the Shallow model

1.5 Figure 4b

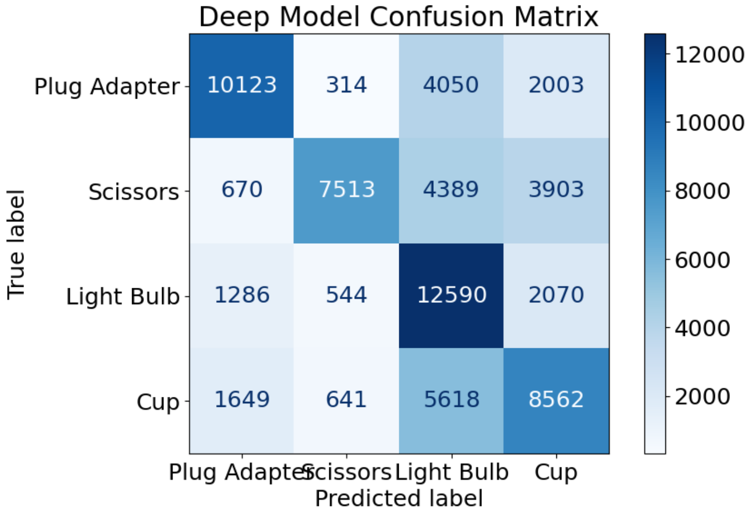

Fig. 5: Confusion Matrix of the test set data across all rotations for the Deep model

1.6 Figure 5

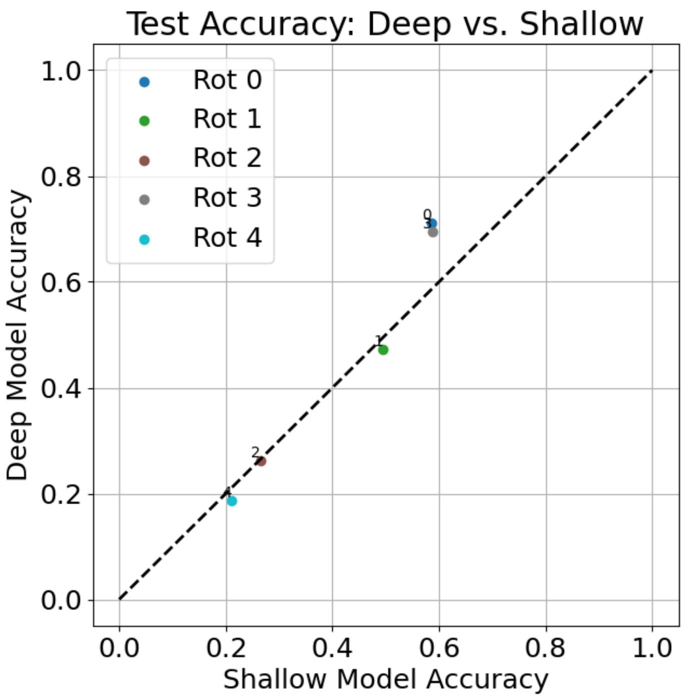

Fig. 6: Test set accuracy for the deep vs shallow networks

2 Analysis & Discussion

"How many parameters were needed by your shallow and deep networks?"

The deep networks have a lot more parameters than the shallow one. By printing the model summary, I have for the shallow network, 3,444 parameters. For the deep network, I have 42,772 parameters.

"What can you conclude from the validation accuracy learning curves for each networks?"

The validation accuracy curve shows that the deep model performs better than the shallow one. However, for the rotation number 2, 3 and 4 both models overfit. We can see that with the validation loss increasing and the training loss decreasing faster than rotations which generalize better.

I think that overfitting is due to the lacking of data samples. The number of samples used for training the model does not represents well the data distribution in some rotation and so the model hardly classifies items that it saw just few times during the training process.

"How confident are you that you have created models that you can trust?"

Models which have trained on rotation 0 tend to be more reliable because the accuracy and the loss of the testing and validation sets are pretty good. However, I would not be confident because there are scenarios where training is hard that the model is really bad. In real life, there are situtations that some images, samples from a class is really hard to get.

A well-performing model should have stable validation accuracy across different rotations. If deep networks generalize well to test data, they are more reliable.

"Did your shallow or deep network perform better with respect to the test set?"

Where we have really bad data distribution in rotation 2 and 4, the shallow model performs better because it generalizes more than the deep model. However, when the data distribution is "good" such as rotation 0 and 3, the deep model is way better than the shallow model.

"Describe the errors that your shallow and deep networks tend to make."

We can see with figure 3, figure 4a and figure 4b that the shallow model has difficulties to "clearly" classify an object. There are always some probabilities that the object could be from a different class. The deep model, however, tends to clearly classify (probability distribution tends to be 1 for one class). This is good when the models is well-trained on a well distributed train data. However, becase the deep model clearly classify, when it got wrong, it got really wrong and there are no probability distribution to other class to help a post process to identify the good type of objects.

We can see this on the confusion matrices, with the shallow one, the probability are more evenly distributed on all cases, but for the deep confusion matrix, the probability are distributed along the diagonal.

"Is there consistency in the performance in the five runs for your deep network?"

The deep model shows variation across different rotations but remains consistently better than the shallow model. The deep network's points are closer together, showing less variance in results.

As discussed before, when the training data is well distributed, the model performs well and better than the shallow one. However, with some rotations, the models perform really bad.

This is probably because we don't have enough data to train the models. I would increase the dataset. I think we can do it because the object that we want to detect are pretty common.